# UMA NOTA SOBRE SOLUÇÃO PARALELA DE SISTEMAS LINEARES ESPARSOS

L. A. Terry, M. J. Teixeira e S. P. Roméro
CEPEL – Centro de Pesquisas de Energia Elétrica
C.P. 2754
20001 – Rio de Janeiro – RJ

## RESUMO

A solução de um sistema de equações lineares esparso é central para inúmeras aplicações utilizadas no planejamento e operação de sistemas elétricos. Este artigo aborda o problema da solução direta de sistemas lineares esparsos em arquiteturas baseadas em memória compartilhada. O grafo de dependências entre as tarefas envolvidas na solução é apresentado e analisado utilizando exemplos do sistema elétrico brasileiro. Uma implementação paralela, destinada a uma avaliação preliminar dos algoritmos e determinação dos limites do "hardware" disponível no Cepel, também é descrita e os seus resultados são apresentados. Desenvolvimentos destinados a melhorar o desempenho dos algoritmos são sugeridos.

## ABSTRACT

The solution of sparse sets of linear equations is central to numerous applications used in power systems operating and planning. This paper concerns the direct solution of sparse linear equations systems in shared memory architectures. The precedence graph of tasks involved in the solution is presented and analyzed using examples from the brazilian electric power system. A parallel implementation, designed for a preliminary evaluation of algorithms as well as a determination of limitations in the hardware available in Cepel, is also described and its results are presented. Developments to improve the algorithm's performance are suggested.

## 1. INTRODUÇÃO

O emprego de múltiplos microprocessadores operando em paralelo constitui uma alternativa econômica potencial tanto para viabilizar a solução de diversos problemas de planejamento e operação de sistemas elétricos, como para melhorar o desempenho dos algoritmos utilizados atualmente ([1],[5]).

Em [1], os problemas onde se pode aplicar o processamento paralelo foram classificados em "fracamente acoplados" e "fortemente acoplados" segundo a relação entre o volume de informações trocadas pelas tarefas executadas e a quantidade de processamento realizado independentemente por cada uma delas. Algumas aplicações "fracamente acopladas" na área de sistemas elétricos, foram também apresentadas ali. Uma destas aplicações, o "despacho de potência com restrições de segurança", é analisada em outro artigo [7], onde são apresentados resultados práticos reais obtidos no CEPEL com a utilização de uma arquitetura multi–processadores com 16 módulos iAPX286/287 PP (Processador Preferencial) do CPqD/TELEBRÁS.

O presente trabalho retoma o tema das aplicações "fortemente acopladas", em que a comunicação é grande relativamente ao processamento local. Em sistemas elétricos, aplicações deste tipo são, por exemplo [6], o cálculo de fluxo de potência, a simulação de transitórios eletromecânicos, o cálculo de curto–circuitos e a simulação de transientes eletromagnéticos.

Cada uma dessas aplicações, computacionalmente intensivas de per si, pode ser muitas vezes utilizada como módulo básico para construir aplicações mais complexas, porém em geral fracamente acopladas, como mencionado em [1]. É o caso do fluxo de potência, utilizado como sub–problema da análise de contingências no "despacho com restrições de segurança" [7]. Algumas delas, entretanto, são de tal modo exigentes em termos de capacidade de processamento (como a simulação de transitórios eletromecânicos, por exemplo) que sua própria utilização como módulo básico dessas novas aplicações fracamente acopladas dependerá de tempos de computação significativamente inferiores aos atualmente obtidos com processamento sequencial.

A característica de "acoplamento forte" das aplicações básicas acima, quando se tenta distribuir o processamento por diferentes processadores, vem da necessidade do cálculo conjunto de toda a rede. A rede elétrica pode ser visualizada como um grafo em que os nós são as usinas geradoras e as subestações de fornecimento de energia às redes de distribuição (cidades e bairros) e os ramos são as linhas de transmissão e os transformadores (ver figura 1–a). Tipicamente as redes podem ter até 3000 nós e 5000 ramos (1800 nós e 3000 ramos no sistema interligado Sul/Sudeste brasileiro).

Em síntese, o cálculo da rede pode ser reduzido à solução de sistemas de equações lineares esparsas de grande porte do tipo Ax=b , onde cada incógnita ou equação está associada a um nó de rede. Fora da diagonal, a matriz A tem poucos elementos não–nulos, que estão por sua vez associados aos ramos da rede.

A figura 1–b representa a matriz A correspondente ao diagrama da figura 1–a, sendo indicados por um "x" apenas os elementos não nulos da mesma; nota–se que ela é simétrica em estrutura. A existência de um grande número de elementos nulos em A se deve às poucas interligações que cada nó tem com os demais nós da rede.

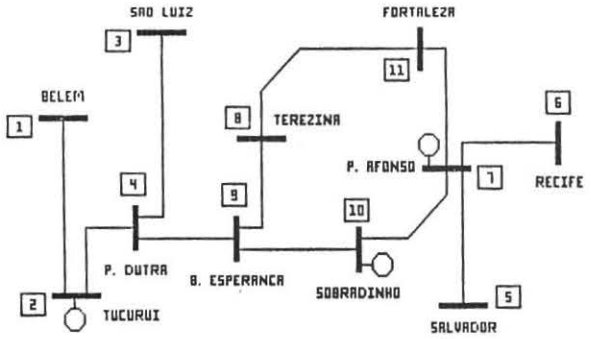


Figura 1–a

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | × | × | | | | | | | | | |
| 2 | × | × | | × | | | | | | | |
| 3 | | | × | × | | | | | | | |
| 4 | | × | × | × | | | | | × | | |
| 5 | | | | | × | | × | | | | |
| 6 | | | | | | × | × | | | | |
| 7 | | | | | × | × | × | | | × | × |
| 8 | | | | | | | | × | × | | × |
| 9 | | | | × | | | | × | × | × | |
| 10 | | | | | | | × | | × | × | |
| 11 | | | | | | | × | × | | | × |

Figura 1–b

No cálculo de fluxo de potência, as equações não–lineares do sistema em regime permanente são resolvidas iterativamente pela solução sequencial de diversos problemas lineares deste tipo. No cálculo de curto–circuitos, é necessário resolver três sistemas lineares deste tipo (um para cada componente simétrica de seqüência), para cada ponto de defeito. Na simulação de transitórios eletromecânicos, os geradores dão origem a equações diferenciais que são integradas numericamente, o que pode ser feito facilmente em paralelo; a cada intervalo de integração, porém, é necessário resolver um sistema de equações lineares esparsas para representar as interações dinâmicas entre as máquinas através da rede. A simulação de fenômenos transientes eletromagnéticos pode ser feita integrando–se numericamente no tempo as equações diferenciais de ondas devidamente discretizadas; isto dá origem a um sistema de equações lineares esparso, correspondente à estrutura da rede, que precisa ser resolvido a cada intervalo de integração. Outra alternativa consiste em trabalhar no domínio da freqüência; neste caso, são resolvidos sistemas lineares esparsos correspondentes ao regime permanente de cada freqüência de um espectro, cujas soluções são posteriormente convoluídas para a obtenção da resposta no tempo.

Vários algoritmos paralelos de solução direta de sistemas lineares esparsos têm sido propostos ([2],[3],[4]). Métodos iterativos de solução também estão sendo tentados. Entretanto, ainda não existem conclusões definitivas quanto aos algoritmos mais adequados para a solução destes problemas. O padrão irregular da estrutura da matriz esparsa não permite aproveitar o potencial de cálculo de "array processors" e dificulta a utilização de arquiteturas distribuidas como as baseadas em arranjos geométricos de processadores, por exemplo, hipercubos. Idealmente, uma arquitetura baseada em uma memória compartilhada por todos os processadores (se puderem ser contornados os problemas de contenção) seria a mais indicada, tanto sob o ponto de vista de facilidade de programação quanto pelo de desempenho potencial do algoritmo.

Este artigo estuda o problema da solução de sistemas lineares pelo método direto (eliminação de Gauss), utilizando casos–exemplo do sistema elétrico brasileiro. Como no caso de sistemas elétricos é normalmente necessário resolver uma série de problemas lineares com a mesma matriz A de coeficientes, isto é, variando apenas o termo independente b, analisou–se prioritariamente o processamento paralelo das substituições sobre o termo independente, não se tendo ainda considerado algoritmos paralelos para a decomposição de A em seus fatores triangulares. São apresentados resultados de simulações e de uma implementação feita no multiprocessador com 16 UCP's tipo PP. Esta implementação objetivou determinar os limites atuais e a eventual adequação desta arquitetura à solução de problemas fortemente acoplados.

## 2. DESCRIÇÃO DO PROBLEMA

O método de eliminação de Gauss para a solução de Ax = b [6] envolve a representação de A como o produto de 3 matrizes : A = LDU, onde L e U são matrizes triangulares, respectivamente inferior e superior e D é uma matriz diagonal. Os fatores L, D e U são determinados por operações simples de pivoteamento entre as linhas da matriz A. A figura 2 ilustra a decomposição L D U da matriz A para o sistema da figura 1–a.

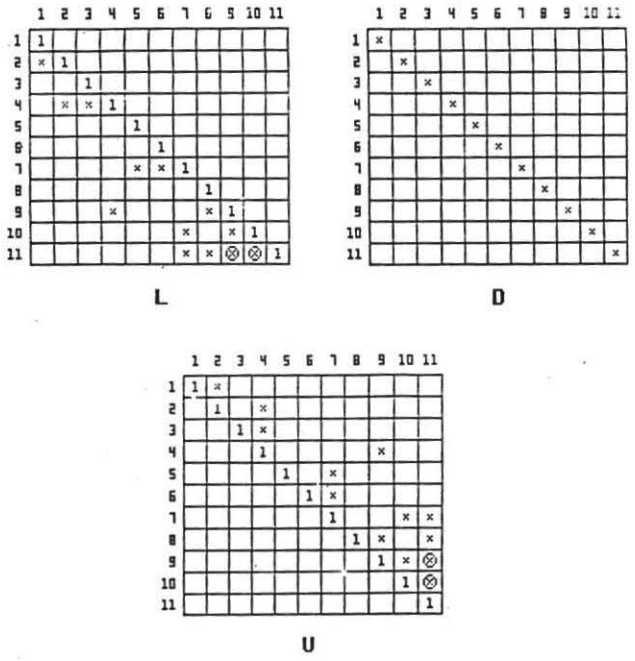


Figura 2

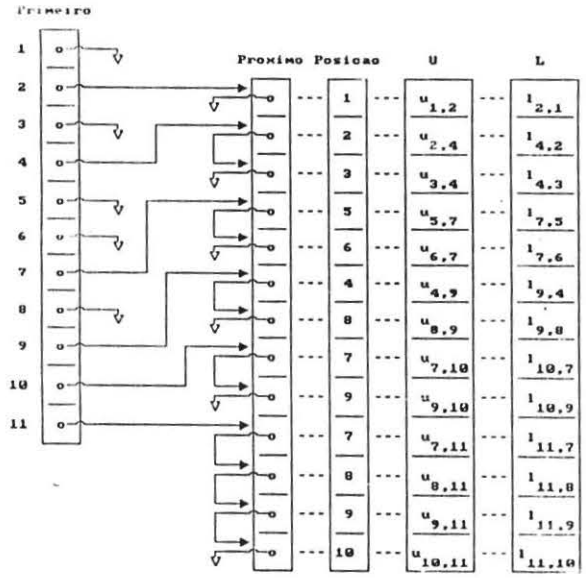


Figura 3

Uma vez obtidos os fatores, os passos para a determinação de uma solução para um vetor independente b são :

1. resolver em z  Lz = b
2. resolver em y  Dy = z
3. resolver em x  Ux = y

o que equivale a resolver L D U x = b ou A x = b. Na prática os vetores z, y e x são calculados sobre o próprio vetor b. A solução destes sistemas, reduz–se à realização de uma seqüência de operações aritméticas elementares sobre o termo independente b. O passo 1 será chamado de substituição progressiva ou fase L, o passo 2 de normalização ou fase D e o passo 3 de substituição regressiva ou fase U.

A grande vantagem da eliminação de Gauss para a solução de sistemas lineares esparsos reside em que, se a ordem das operações de pivoteamento for escolhida com cuidado, as matrizes L e U resultam também esparsas. Na figura 2 pode ser observado que foram criados apenas dois novos elementos não–nulos em L e em U ($l_{11,9}$ e $l_{11,10}$ em L e seus simétricos em U). Em conseqüência da esparsidade, tanto a matriz A, inicialmente, como depois as matrizes L e U podem ter armazenados apenas os seus elementos não nulos, usando–se, por exemplo, estruturas de listas encadeadas, como as da figura 3.

Em princípio, a preservação da esparsidade é conseguida observando–se duas regras ao longo da triangularização :

1. os elementos pivôs nas operações de eliminação são escolhidos sempre na diagonal, o que para uma matriz A simétrica em estrutura preserva a simetria dos elementos dos fatores L e U e simplifica as listas encadeadas necessárias.

2. a cada passo do processo, o pivô sempre é escolhido na linha que contém o menor número de elementos não nulos. Este é um critério heurístico que, pela sua eficiência em miniminizar a criação de elementos não nulos adicionais, tornou–se um padrão utilizado tradicionalmente nos algoritmos esparsos seqüenciais.

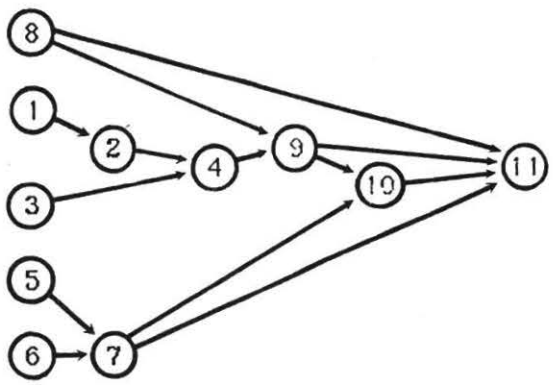


Figura 4–a

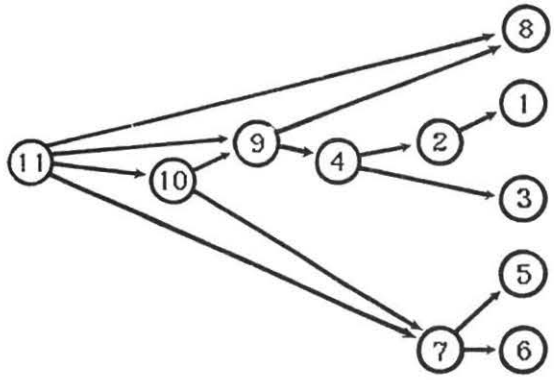


Figura 4–b

Assim como a matriz de coeficientes A está associada ao grafo do sistema elétrico (figuras 1–a e 1–b), é possível associar aos fatores L e U dois grafos orientados, cujos ramos correspondem aos seus elementos não nulos. Estes grafos indicam a precedência com que devem ser realizadas as operações de substituição, progressiva ou regressiva, e são chamados aqui de "grafo de tarefas" das fases L ou U, respectivamente. As figuras 4–a e 4–b mostram estes grafos para o caso da triangularização da matriz da figura 1–b, observadas as regras 1 e 2 acima. Consequência da simetria estrutural dos fatores L e U, o grafo de tarefas da fase U é igual ao da fase L, porém com os ramos orientados no sentido inverso.

Nos grafos de tarefas, os ramos que partem de um nó podem ser liberados para serem processados, logo após o processamento de **todos** os ramos que nele incidem :

- Durante a fase L, cada componente $b_i$ do vetor independente fornece contribuições para outros componentes $b_j$, de modo que $b_j := b_j - l_{ji}b_i$. Cada $b_i$ fornecerá tantas contribuições quantos forem os elementos $l_{ji}$ existentes na coluna i de L. Correspondentemente, no grafo de tarefas da fase L cada contribuição de $b_i$ para $b_j$ é representada por um ramo orientado do nó i para o nó j. Um termo $b_i$ só pode fornecer contribuições a outros depois que já tiver recebido todas as contribuições, relativas aos elementos $l_{ik}$ existentes na linha i de L (ramos incidentes no nó i do grafo).
- A fase D consiste nas normalizações dos componentes $b_i$ que são da forma $b_i := b_i/d_{ii}$, onde $d_{ii}$ é um dos elementos do fator D.
- Na fase U, cada componente $b_i$ fornecerá contribuições do tipo $b_j := b_j - u_{ji}.b_i$, correspondentes aos elementos $u_{ji}$ existentes na coluna i do fator U.

O critério heurístico utilizado na triangularização ordenada de sistemas esparsos visa diretamente a economia de memória mas tem também um efeito decisivo na redução do esforço computacional, em conseqüência da esparsidade conseguida nos fatores L e U. Uma outra constatação importante, feita a partir dos grafos de tarefas das fases L e U, é a de que ele induz também a notáveis oportunidades de paralelismo, especialmente nos estágios iniciais da fase L e nos finais da fase U.

Nos sistemas conexos existe apenas um nó terminal no grafo de tarefas da fase L, que corresponde também ao único nó inicial da fase U. Isto impede que substituições das duas fases possam ser realizadas simultaneamente. As normalizações da fase D, por outro lado, podem ser executadas assim que cada nó na fase L recebe todas as contribuições de seus antecessores. Se todas as possibilidades de paralelismo puderem ser exploradas, o tempo total mínimo para a solução paralela de um sistema esparso será igual ao tempo necessário para a fase L, mais o tempo de normalização do nó terminal e mais o tempo da fase U (igual ao da fase L).

Em cada uma das fases L e U, utilizando–se um número ilimitado de processadores, o tempo de processamento será igual ao tempo do caminho crítico dos grafos de tarefas respectivos. Considerando um mesmo tempo unitário para o processamento de cada ramo dos grafos L ou U, definem–se:

. Tempo cedo do nó i = TC(i) =

$$= \begin{cases} \max_j\{TC(j) \mid j \text{ antecessor de } i\} + 1 \\ 0 \text{ se } i \text{ não tem antecessor} \end{cases}$$

Tempo tarde do nó i = TT(i) =

$$= \begin{matrix} A\,min_j\{TT(j) \mid j \text{ sucessor de } i\} - 1 \\ B\,TC(i) \quad \text{se } i \text{ não tem sucessor} \end{matrix}$$

O caminho crítico corresponde à seqüência de tarefas ligando nós i com TC(i) = TT(i).

## 3. RESULTADOS DA SIMULAÇÃO

Foram realizadas simulações de soluções de sistemas lineares esparsos utilizando fatores L D U para duas redes elétricas brasileiras com respectivamente 470 e 1557 barras. A triangularização ordenada da matriz destes sistemas leva a fatores triangulares com, respectivamente, 1123 e 3347 elementos não nulos, ou a grafos de tarefas com estes números de ramos. (Estes números expressam também a duração total de um processamento sequencial de cada uma das fases L e U).

As simulações foram feitas admitindo–se que :

- não existem outras contenções (de "hardware", por exemplo) que não as decorrentes das próprias precedências das tarefas.
- cada tarefa elementar, das fases L, D e U possui duração unitária.
- as tarefas da fase L têm precedência sobre as

da fase D (quando estas já podem ser executadas). As tarefas da fase D só são executadas se, em algum instante, houver um processador ocioso.

- a cada instante nas fases L e U, os processadores executam prioritariamente, dentre as substituições que já podem ser realizadas, aquelas cujos ramos incidem em nós com os menores tempos tarde (TT).

A tabela 1 apresenta os dados das simulações para os dois sistemas testados, para a condição de aproveitamento máximo do paralelismo.

Tabela 1

| | A | B |
|---|---|---|
| Número de nós | 470 | 1557 |
| Elementos não nulos em L ou U | 1123 | 3347 |
| Tempo solução com 1 processador | 2716 | 8251 |
| Tempo mínimo de solução paralela | 65 | 93 |
| Melhoria máxima ("speed–up") | 41.8 | 88.7 |
| Número limite de processadores | 49 | 105 |
| Eficiência (%) | 85.3 | 84.5 |

Na realidade, os "speed–ups" e eficiências acima precisariam ser reduzidos por um fator, dependente do "hardware" e estimado em cerca de 25% para o iAPX286, para levar em conta que um programa puramente sequencial não necessita de operações de sincronização e que as operações de normalização podem ser feitas em um tempo menor que as das fases L e U.

Na figura 5 é mostrada a melhoria ("speed–up") para a solução dos dois sistemas, em função do número de processadores. É interessante observar que a melhoria cresce linearmente enquanto o número de processadores é pequeno, saturando ao atingir o número limite de processadores da tabela 1.

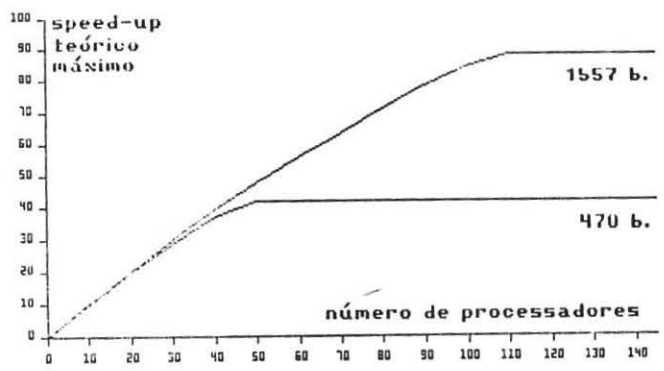


Figura 5

O tempo mínimo de solução paralela está associado à estrutura do grafo de tarefas. Para minimizá–lo seria preciso empregar durante a triangularização um critério de ordenação com este objetivo, já que o critério utilizado nos casos acima objetiva apenas minimizar a criação de novos elementos não nulos. A obtenção de tal ordenação é um problema de difícil tratamento [2].

Em função disto, foi experimentada uma modificação visando diminuir o tempo de solução paralela, sem contudo se afastar do critério tradicional de ordenação que tende a minimizar o número de tarefas nos grafos. Ela consiste em escolher–se o pivô na linha da matriz (nó no grafo de tarefas) com o menor tempo cedo, dentre aquelas com um mesmo menor número de elementos. Esta modificação, de fácil implementação pois o tempo cedo de uma linha da matriz é função apenas do tempo cedo das que já foram pivoteadas anteriormente, retarda a eliminação de linhas (nós) com tempo cedo alto, e colabora para a redução do tempo mínimo de solução paralela do sistema. Em alguns sistemas esta modificação pode ser importante. No sistema teste padrão "IEEE 118 barras" ela resulta numa redução de aproximadamente 40% do tempo mínimo de solução e num aumento de 60% no "speed–up" máximo.

## 4. RESULTADOS EXPERIMENTAIS

No item anterior foram feitas hipóteses simplificadoras sobre a execução das tarefas elementares em ambiente paralelo. Em uma implementação real, fatores como a contenção do barramento tornam–se primordiais, impondo sérias limitações à eficiência prática dos algoritmos paralelos. A determinação dos limites atuais do PP na solução de problemas fortemente acoplados foi uma das motivações centrais desta implementação.

Na implementação efetuada atribui–se um conjunto de tarefas elementares a cada um dos processadores. À diferença de [3], a distribuição das tarefas é estática (i.e., cada processador recebe antes de começar a solução do sistema a lista de quais tarefas deverá executar) e é feita durante a fatoração da matriz. A distribuição estática oferece duas vantagens :

- os fatores LDU podem ser armazenados na memória interna de cada processador, evitando acessos à memória comum;
- evita–se o processo de escolha dinâmica de tarefas, que é dispendioso em termos de sincronização.

Cada elemento de um fator obtido durante a fatoração é atribuido a um dos processadores que, durante a solução, executará sua operação associada. A cada processador são alocadas prioritariamente as tarefas que incidem em nós com menor tempo tarde (definido no item 3). Procura–se também distribuir estas tarefas de maneira a que todos os processadores tenham um número aproximadamente igual de operacões a realizar. Ao final da etapa de fatoração, cada processador possui localmente uma lista de

tarefas para cada uma das fases L, D e U, nesta ordem. A cada passo, o processador se não puder executar a próxima operação prevista, entrará em espera ocupada até poder realizá–la.

Em memória global foram colocados: o vetor independente (x), semáforos contadores para sincronização entre tarefas em cada uma das fases L e U, e semáforos binários para cada um dos elementos do termo independente. Em cada fase, os contadores ocupam um vetor de n posições – onde n é a dimensão do termo independente – e, ao término de cada substituição, é decrementada a posição correspondente no vetor. Na fase L estes contadores têm a dupla função de dizer se um elemento do termo independente pode contribuir para outros e se este elemento pode ser normalizado; os da fase U têm a função de dizer se um elemento do termo independente já foi normalizado e se já pode contribuir para outros.

A implementação foi executada no PP em Fortran, com o acréscimo de rotinas em assembler para os serviços de semáforos na memória global.

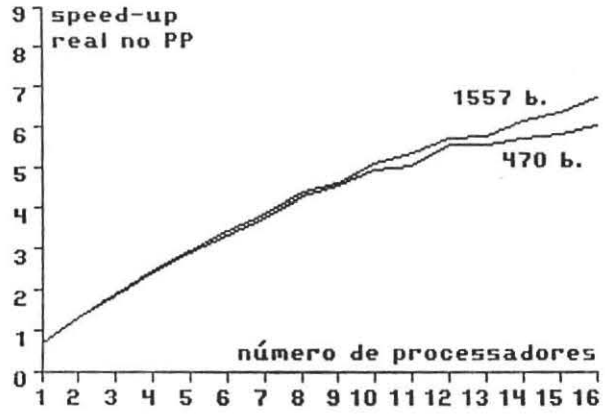


Figura 6

A figura 6 mostra os "speed–ups" reais obtidos com esta implementação em relação a uma versão seqüencial. Mesmo levando em conta o "over–head" antes estimado de 25% para o iAPX286, os resultados são inferiores aos previstos no capítulo anterior e também em [3], quando várias dezenas de processadores ainda permitiam eficiências elevadas. A razão disso é a contenção no acesso aos dados da memória compartilhada do PP. Algumas melhorias no "hardware" do PP, neste sentido, permitiriam a utilização eficiente de um número maior de processadores, respeitados os limites teóricos deste tipo de aplicação. Estas modificações seriam em ordem de importância: a substituição da memória comum (hoje duas vezes mais lenta que as memórias locais) por uma mais veloz, o emprego de um esquema de arbitragem centralizada dos acessos ao barramento e a adoção de um barramento mais rápido ou outro mecanismo que assegure maior velocidade no acesso aos dados da memória compartilhada.

## 5. OTIMIZAÇÃO DA PERFORMANCE

Uma maneira de otimizar os algoritmos de solução direta em arquiteturas com memória comum é a distribuição de um conjunto de tarefas para cada processador, de maneira a minimizar a comunicação exigida entre processadores. O problema de atribuição de tarefas de duração unitária, cujo grafo de dependência é uma árvore, é NP–completo [8], o que inviabiliza na prática a determinação de soluções ótimas para o problema estudado. Esta dificuldade poderia ser contornada com a proposição de heurísticas que conseguissem particionar o grafo em sub–grafos que tenham pouca ou nenhuma comunicação entre si. Isto equivale a alocar sub–vetores do vetor b localmente a cada processador, reduzindo o número de acessos à memória comum. Estas heurísticas devem possuir, entretanto, regras razoavelmente simples para não degradar a performance da solução como um todo (montagem da matriz, fatoração e solução do sistema) e não parece ser tarefa fácil encontrá–las.

Uma outra idéia, mais simples, e que está em fase de implementação, consiste em atribuir a um mesmo processador, na fase L e na fase U, todas as tarefas que possuem arcos incidindo num determinado nó do grafo. No esquema atualmente implementado, o tempo total despendido em leituras, escritas e acessos a semáforos em memória global é de aproximadamente $2l(2L+E+S)+n(L+E)$ , onde n é o número de nós do grafo de tarefas (L ou U), l o seu número de arcos, e L,E,S os tempos de uma leitura, de uma escrita e de um acesso a semáforo, respectivamente. Estas são as tarefas que criam contenção no acesso à memória comum. Para a atribuição de tarefas proposta, este número se reduz a $2lL+2n(L+E)+n(L+E)$. Eliminam–se portanto os semáforos binários antes necessários à atualização dos elementos dos termos independentes, além de diminuir de $2(l-n)(E+L)$ o tempo de uso da memória compartilhada.

Como já mencionado, um aspecto que pode ser atacado de imediato visando uma melhoria do desempenho de aplicações fortemente acopladas é o "hardware":

- a atual utilização de uma memória compartilhada lenta no PP agrava o problema de contenção, levando a uma queda na eficiência do algoritmo bem antes de seu limite teórico;
- outro ponto que certamente traria melhorias é a substituição do esquema "daisy–chain" circular de arbitragem de acesso ao barramento por um árbitro centralizado;
- modificações mais profundas no "hardware" poderiam incluir ainda, prioritariamente, mecanismos mais eficientes no acesso à memória compartilhada. Possibilidades neste sentido são, por exemplo, o emprego de barramentos mais rápidos, a multiplexação dos acessos à memória, a adoção de dispositivos do tipo "cross–bar", ou a utilização de chaves digitais para o roteamento dos acessos.

A utilização do processador iAPX386, prevista no novo projeto do PP, trará uma melhoria apreciável de desempenho. Está também prevista a substituição do co–processador numérico 80287 pelo 80387 ou WTL–1167, o que permitirá melhor desempenho em aplicações numericamente intensivas. Para aplicações fortemente acopladas, o emprego destes processadores mais rápidos imporá, naturalmente, maiores exigências ao desempenho da memória compartilhada e de seus mecanismos de acesso.

## 6. CONCLUSÃO

Resultados de simulação e da implementação efetuada mostram ser possível obter ganhos consideráveis no desempenho de algoritmos de solução por método direto de sistemas lineares esparsos através do emprego de máquinas paralelas com memória comum e algumas dezenas de processadores. O número adequado de processadores depende de detalhes do "hardware" e da implementação do algoritmo. Arquiteturas deste tipo parecem ser as mais indicadas pela facilidade oferecida para a programação e performance potencial dos algoritmos. É preciso, entretanto, que seja ainda despendido um esforço de pesquisa no sentido de se obter memórias e barramentos mais rápidos. É importante também pesquisar refinamentos nos algoritmos existentes de maneira a viabilizar a solução paralela de problemas fortemente acoplados, centrais em sistemas de energia elétrica.

## 7. REFERÊNCIAS


[1] Pereira M.V.P., Teixeira M.J. e Terry L.A., "Aplicações de processamento paralelo em sistemas elétricos de potência", 1° Simp. Bras. Proc. Paralelo, Gramado, RS, maio 1987.

[2] Wing O. e Huang J.W., "A computation model of parallel solution of linear equations", IEEE Tr. Comp., vol.C–29, no.7, pp. 632–638, julho 1980.

[3] Arnold C.P., Parr M.I. e Dewe M.B., "An efficient parallel algorithm for the solution of large sparse linear matrix equations", IEEE Tr. Comp., vol.C–32, no.3, pp.265–273, 1983.

[4] Van Ness J.E., "Multiple factoring in the parallel solution of algebraic equations", relatório EPRI EL–3893, março 1985.

[5] Teixeira M.J., Pereira M.V.P., Terry L.A. e Pinto H.J.C.P., "Ambiente para desenvolvimento de programas paralelos fracamente acoplados", 7° Cong. SBA, julho 1988.

[6] Monticelli A.J., "Fluxo de carga em redes de energia elétrica", Ed. E. Blucher, 1983.

[7] Teixeira M.J., Pinto H.J.C.P. e Pereira M.V.P., "Despacho com restrições de segurança e controle corretivo – uma implementação paralela", submetido ao 2° Simp. Bras. Arq. Comp., Lindoia, SP, 1988.

[8] Carlier J. e Chretienne P., "Un domaine tres ouvert: les problemes d'ordonnancement", RAIRO Op. Research, vol 16, no.3, pp. 175–217, agosto 1982.